INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

# PROGRAMMING WITH PYTHON

## WRITTEN ASSIGNMENT

COURSE NAME: ADVANCED CREDIT COURSE FOR MASTER IN COMPUTER SCIENCE (120 ECTS)

NAME: EVETTE MARK

MATRICULATION NUMBER: UPS10640057

TUTOR: DR. COSMINA CROITORU

**TABLE OF CONTENTS:**

**ABSTRACT:**

This research work is done to deal with problems related to real-life scenarios where huge sets of data are used daily. Collecting, transforming, cleaning, storing, standardizing, processing are all hefty tasks to be carried out manually especially because of the huge volume of the data that needs to be worked upon here. Different companies require different information related to their clients to analyze it for various purposes.

The final domain-related data is used as was intended like for an e-commerce website, it can be to manage demand and supply, or to study each client's ordering habits to provide them with appropriate suggestions for further purchases. Doing it manually would have costed a lot more money, time and resources than having a Python program do it all for you in a matter of seconds to minutes.

There are many more useful and common scenarios where regression is applied regularly. Prediction of prices of all sorts of commodities from foodgrains to houses, forecast of sales revenue, healthcare costing, equipment failures, student performance and so much more is done using their various factors like season, weather, demand, occasion, time of day, and so on for regressions.

## INTRODUCTION:

One of the most important issues these days is the increase in traffic and traffic congestions due to people buying their own vehicles instead of using public transports or walking. Although this gives the people a sense of control of their time, if not routed properly, they might end up wasting it in unnecessary traffic jams. In order to resolve this issue, traffic flow prediction is important. We can find it being applied on Google Maps.

Many a times we end up reaching places with dug up roads or blocked pathways due to construction, processions, accidents and so on. The assignment will be focusing on the part where data for 100 days is recorded on the busiest of days, the rainy season, while recording the busiest of streets. This will help us get a better estimation of traffic flow on a daily basis so that the best routes can be chosen instead.

Data Set

A collection of related information sets that are made up of separate elements but can be manipulated as a unit by a computer. The dataset presented here has 3 parts, namely the ideal, the train and the test type. Data for this program is saved in the CSV format.

The ‘ideal’ dataset consists of 1 value of x with 50 corresponding values of y.

The ‘train’ dataset has 1 value of x and 4 corresponding values of y.

The ‘test’ dataset is the one that we require with 1 value of x and its corresponding y.

## Problem Statement

Need to describe a scenario where 50 ideal situations / datasets will be compared. From these 50, we require selecting 4 training datasets that fit perfectly for the situation chosen to train the dataset to select 1 testing dataset from among them so that we can reach an appropriate estimation for prediction of future scenarios. For completing the mentioned steps, we will be making use of the least-squared type of analysis in regression.

In the additional task, we will be adding the proof of the above on a GitHub account so that the program can be tested and used with different datasets. It will have the datasets, the code in the Jupyter Notebook.

## Goal

To create a program that handles a lot of data with accordance to the purposes that a company needs to cater. A huge amount of data; thought to be ideal for training, needs to be sifted first to shortlist the best datasets for training the program to familiarize it with possible scenarios that the model might encounter, to carefully come up with calculated solutions in real-time. These

shortlisted datasets are further sifted to choose the best testing dataset for the required purpose. Every program goes through enumerable trainings and tests before being deployed. This helps the program become more efficient and reliable.

All the above is done with the objective to help companies handle datasets in the real-world speedily and to help reduce expenses on human resources.

## Overall Scope

This program uses Python programming language with its packages, namely Pandas, Matplotlib and Seaborn, for data science to a vast extent to break it down

into manageable pieces of information that are expressive in structure while being easy to understand and intuitive.

**Overview of few topics**

**Regression:** Relationship between variables of a dataset.

**Package:** A set of classes and interfaces that are all related to one another.

**GitHub:** A cloud service as well as a website that helps in the storing, managing, tracking and controlling changes in a project.

**Jupyter Notebook:** A web application to create and share documents with live code, equations, visualizations and text.

## DATA VISUALIZATION:

The datasets used for this program are vast. To understand them better, the following figures have been displayed. It will help with relating to how it works at the backend.

The Ideal Dataset –

```
In [31]: ideal
```

Out[31]:

| | x | y1 | y2 | y3 | y4 | y5 | y6 | y7 | y8 | y9 | ... | y41 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | -20.0 | -0.912945 | 0.408082 | 9.087055 | 5.408082 | -9.087055 | 0.912945 | -0.839071 | -0.850919 | 0.816164 | ... | -40.456474 | 40.2 |
| **1** | -19.9 | -0.867644 | 0.497186 | 9.132356 | 5.497186 | -9.132356 | 0.867644 | -0.865213 | 0.168518 | 0.994372 | ... | -40.233820 | 40.0 |
| **2** | -19.8 | -0.813674 | 0.581322 | 9.186326 | 5.581322 | -9.186326 | 0.813674 | -0.889191 | 0.612391 | 1.162644 | ... | -40.006836 | 39.8 |
| **3** | -19.7 | -0.751573 | 0.659649 | 9.248426 | 5.659649 | -9.248426 | 0.751573 | -0.910947 | -0.994669 | 1.319299 | ... | -39.775787 | 39.7 |
| **4** | -19.6 | -0.681964 | 0.731386 | 9.318036 | 5.731386 | -9.318036 | 0.681964 | -0.930426 | 0.774356 | 1.462772 | ... | -39.540980 | 39.5 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| **395** | 19.5 | 0.605540 | 0.795815 | 10.605540 | 5.795815 | -10.605540 | -0.605540 | -0.947580 | -0.117020 | 1.591630 | ... | 39.302770 | -38.6 |
| **396** | 19.6 | 0.681964 | 0.731386 | 10.681964 | 5.731386 | -10.681964 | -0.681964 | -0.930426 | 0.774356 | 1.462772 | ... | 39.540980 | -38.8 |
| **397** | 19.7 | 0.751573 | 0.659649 | 10.751574 | 5.659649 | -10.751574 | -0.751573 | -0.910947 | -0.994669 | 1.319299 | ... | 39.775787 | -39.0 |
| **398** | 19.8 | 0.813674 | 0.581322 | 10.813674 | 5.581322 | -10.813674 | -0.813674 | -0.889191 | 0.612391 | 1.162644 | ... | 40.006836 | -39.3 |
| **399** | 19.9 | 0.867644 | 0.497186 | 10.867644 | 5.497186 | -10.867644 | -0.867644 | -0.865213 | 0.168518 | 0.994372 | ... | 40.233820 | -39.5 |

400 rows × 51 columns

*Figure 1: The Ideal Dataset*

The Train Dataset –

```
In [32]: train
```

Out[32]:

| | x | y1 | y2 | y3 | y4 |
|---|---|---|---|---|---|
| **0** | -20.0 | 39.778572 | -40.078590 | -20.214268 | -0.324914 |
| **1** | -19.9 | 39.604813 | -39.784000 | -20.070950 | -0.058820 |
| **2** | -19.8 | 40.099070 | -40.018845 | -19.906782 | -0.451830 |
| **3** | -19.7 | 40.151100 | -39.518402 | -19.389118 | -0.612044 |
| **4** | -19.6 | 39.795662 | -39.360065 | -19.815890 | -0.306076 |
| **...** | ... | ... | ... | ... | ... |
| **395** | 19.5 | -38.254158 | 39.661987 | 19.536741 | 0.695158 |
| **396** | 19.6 | -39.106945 | 39.067880 | 19.840752 | 0.638423 |
| **397** | 19.7 | -38.926495 | 40.211475 | 19.516634 | 0.109105 |
| **398** | 19.8 | -39.276672 | 40.038870 | 19.377943 | 0.189025 |
| **399** | 19.9 | -39.724934 | 40.558865 | 19.630678 | 0.513824 |

400 rows × 5 columns

*Figure 2: The Train Dataset*

The Test Dataset –

```
In [33]: test
```

Out[33]:

| | x | y |
|---|---|---|
| **0** | 17.5 | 34.161040 |
| **1** | 0.3 | 1.215102 |
| **2** | -8.7 | -16.843908 |
| **3** | -19.2 | -37.170870 |
| **4** | -11.0 | -20.263054 |
| **...** | ... | ... |
| **95** | -1.9 | -4.036904 |
| **96** | 12.2 | -0.010358 |
| **97** | 16.5 | -33.964134 |
| **98** | 5.3 | -10.291622 |
| **99** | 17.9 | 28.078455 |

100 rows × 2 columns

*Figure 3: The Test Dataset*

To understand the above in a trend, graphs are depicted as below.

The Ideal Dataset -

```
In [41]: three.figure(figsize=(15,6))
         four.lineplot(x=ideal['x'], y=ideal['y1'], data=ideal)
         three.show()
```

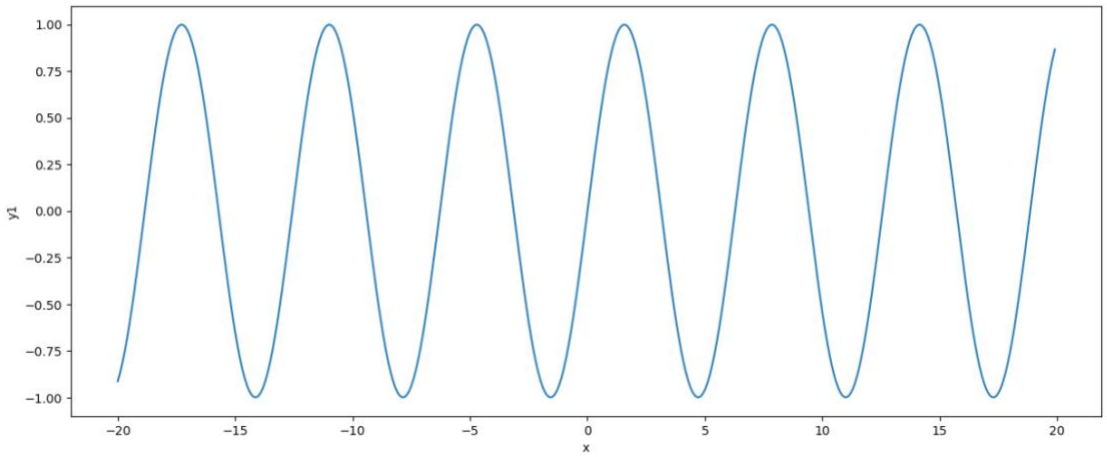


*Figure 4: The Ideal Dataset (2)*

The Train Dataset –

```
In [42]: three.figure(figsize=(15,6))
         four.lineplot(x=train['x'], y=train['y1'], data=train)
         three.show()
```

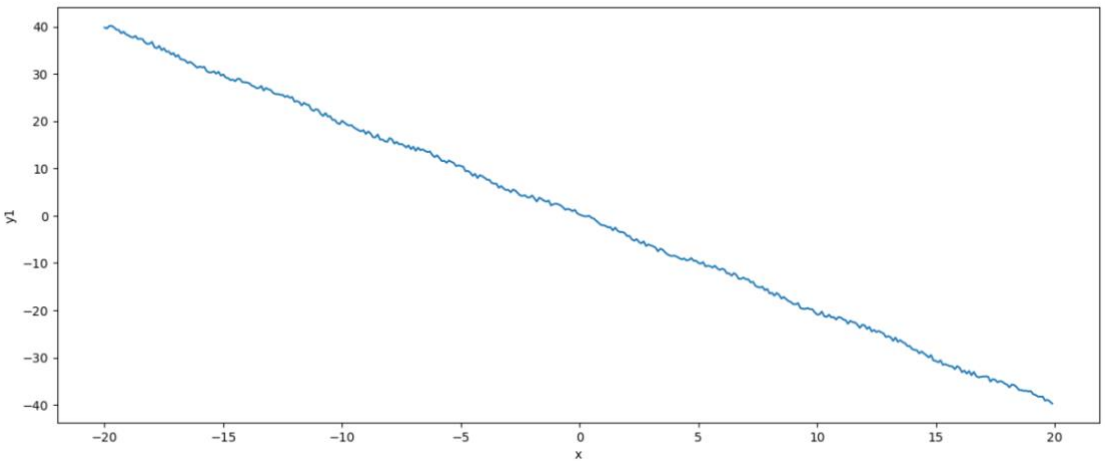


*Figure 5: The Train Dataset (2)*

The Test Dataset –

```
In [44]: three.figure(figsize=(15,6))
         four.lineplot(x=test['x'], y=test['y'], data=test)
         three.show()
```

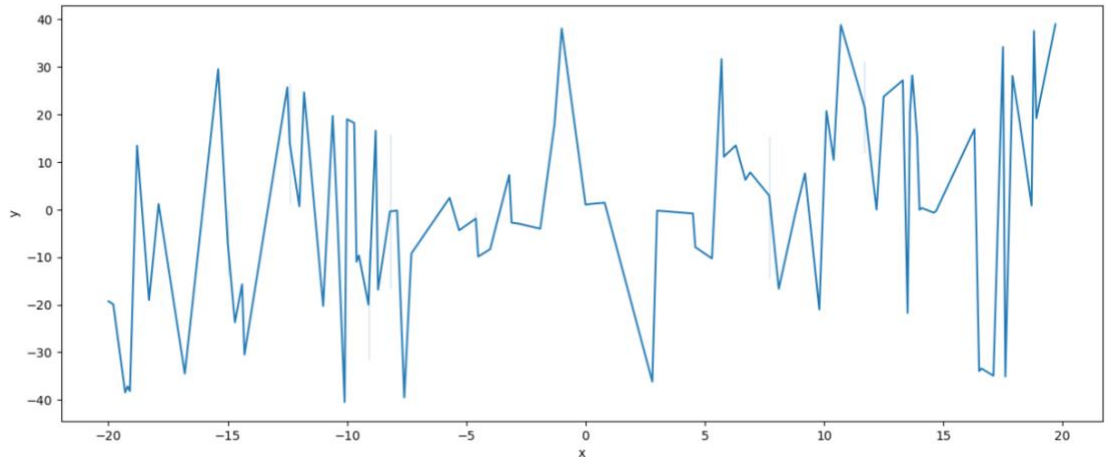


*Figure 6: The Test Dataset (2)*

And finally, them working together in the below displayed Combo Chart -

```
In [61]: three.figure(figsize=(15,6))
         four.lineplot(x=ideal['x'], y=ideal['y1'], data=ideal)
         four.lineplot(x=test['x'], y=test['y'], data=test)
         three.scatter(x=train['x'], y=train['y1'], s=1, c='red')
         three.show()
```

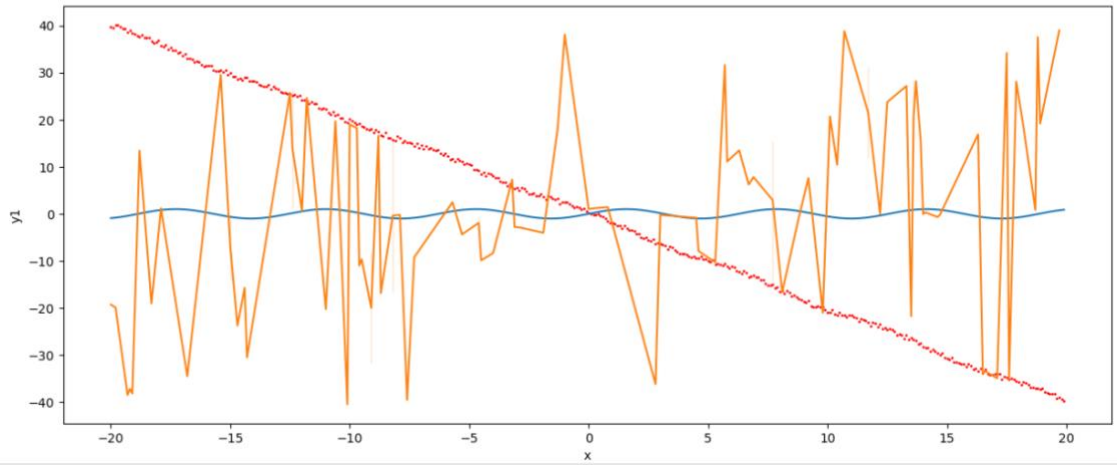


*Figure 7: The Combo Chart*

For the above graphical visualizations, the Python packages, Matplotlib and Seaborn have been used. While Seaborn is a perfect solution for creating plots with minimal code, Matplotlib has more customization options and control over every detail.

Matplotlib is used to create interactive visualizations that are static as well as animated with the help of its comprehensive library. It allows us to work with data in details with features of zooming, panning and updating. This package is used for creating publication quality plots. Here, we have used it with its Pyplot functions to make it work like MATLAB.

Seaborn is also a library that in actuality, works with Matplotlib underneath for plotting graphs. Functions of figure-size, scatter plot for 'train' data in Figure 7, as well as show, that have been used here, are all parts of it.

Observation

The ideal dataset spans only between 1 and -1 while the training as well as test dataset ranged from 40 to -40. In the Combo chart, the ideal and test datasets have been depicted using Line graphs while the train dataset is shown using Scatter plot.

## CONCLUSION:

Dataset of 100 days has been revised using the ideal scenarios from the possible training scenarios and has passed the testing phase. Now the program is ready to make predictions of traffic flow in the most congested areas.

**APPENDIX:**

```
import pandas as one
import numpy as two
import matplotlib.pyplot as three
import seaborn as four
train=one.read_csv("train.csv")
test=one.read_csv("test.csv")
ideal=one.read_csv("ideal.csv")
train.head()
test.head()
ideal.head()
train.shape, ideal.shape, ideal.shape
for j in range(1, len(train.columns)):
    lse2=[]
    for k in range(1, len(ideal.columns)):
        msev=0
        for i in range(len(train)):
            a=train.iloc[i,j]
            b=ideal.iloc[i,k]
            msev=msev+((a-b)**2)
        lse2.append(msev/len(train))
        minmin=min(lse2)
        index=lse2.index(minmin)
        lse2.append(minmin)
```

**ADDITIONAL TASK:**

markevette/DLMDSPW01: Traffic Flow Prediction (github.com)

The aforementioned code and its visualizations can be seen at the link here.